

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号
特開2002-353960
(P2002-353960A)

(43) 公開日 平成14年12月6日 (2002.12.6)

(51) Int.Cl. ⁷	識別記号	F I	テマート* (参考)
H 0 4 L 9/32		G 0 6 F 9/46	3 6 0 B 5 B 0 1 7
G 0 6 F 1/00		12/14	3 2 0 B 5 B 0 3 3
9/318		15/16	6 1 0 Z 5 B 0 4 5
9/46	3 6 0	H 0 4 L 9/00	6 7 5 D 5 B 0 7 6
12/14	3 2 0		6 0 1 C 5 B 0 9 8

審査請求 未請求 請求項の数10 O L (全 18 頁) 最終頁に続く

(21) 出願番号	特願2001-162271 (P2001-162271)	(71) 出願人	000005223 富士通株式会社 神奈川県川崎市中原区上小田中4丁目1番1号
(22) 出願日	平成13年5月30日 (2001.5.30)	(72) 発明者	蒲田 順 神奈川県川崎市中原区上小田中4丁目1番1号 富士通株式会社内
		(72) 発明者	小谷 誠剛 神奈川県川崎市中原区上小田中4丁目1番1号 富士通株式会社内
		(74) 代理人	100074099 弁理士 大菅 義之 (外1名)

最終頁に続く

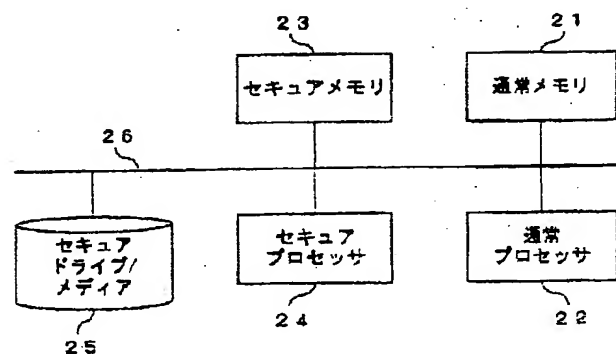
(54) 【発明の名称】 コード実行装置およびコード配布方法

(57) 【要約】

【課題】 既存OSを大幅に変更することなく、電子署名と暗号化が施されたコードを効率よく実行することが課題である。

【解決手段】 セキュアプロセッサ24と通常プロセッサ22を含むヘテロなマルチプロセッサシステムを構築し、セキュアタスクと非セキュアタスクを各プロセッサへ振り分ける。セキュアタスクの暗号化コードは、セキュアメモリ23に格納され、セキュアメモリ23は、認証局の公開鍵で署名を検証して、暗号化コードが有効であることをセキュアプロセッサ24に通知する。セキュアプロセッサ24は、セキュアメモリ23から暗号化命令をフェッチし、復号して実行する。

第1のマルチプロセッサシステムの構成図



【特許請求の範囲】

【請求項1】 マルチプロセッサシステムを用いたコード実行装置であって、セキュアタスクの暗号化コードと、該暗号化コードが正当であることを検証するための検証用情報とを格納するセキュアメモリ手段と、前記検証用情報により前記暗号化コードが正当であると検証されたとき、該暗号化コードを実行するセキュアプロセッサ手段と、通常タスクのコードを格納する通常メモリ手段と、前記通常タスクのコードを実行する通常プロセッサ手段と、前記セキュアタスクと通常タスクの振り分けを行って、前記暗号化コードを前記セキュアメモリ手段に格納し、前記通常タスクのコードを前記通常メモリ手段に格納する制御手段とを備えることを特徴とするコード実行装置。

【請求項2】 前記セキュアメモリ手段は、物理メモリ割り当ての単位毎に前記暗号化コードを格納し、該単位毎の暗号化コードに対する検証用情報を格納し、該検証用情報を用いて該単位毎の暗号化コードを検証し、前記セキュアプロセッサ手段は、正当であると検証された暗号化コードに含まれる暗号化命令をフェッチし、復号して実行することを特徴とする請求項1記載のコード実行装置。

【請求項3】 前記暗号化コードを固有鍵でさらに暗号化して格納するセキュアドライブ手段をさらに備え、該セキュアドライブ手段と前記セキュアメモリ手段は、相互認証を行ってセッション鍵を共有し、該セキュアドライブ手段は、前記制御手段からの読み出し指示に基づいて、該暗号化コードを該固有鍵で復号し、該セッション鍵で暗号化して、該セキュアメモリ手段に転送することを特徴とする請求項1記載のコード実行装置。

【請求項4】 前記セキュアプロセッサ手段は、前記暗号化コードを用いて、該暗号化コードを実行する論理回路の少なくとも一部を、回路状態として不揮発的に固定することを特徴とする請求項1記載のコード実行装置。

【請求項5】 物理メモリ割り当ての単位毎に暗号化コードを格納する手段と、該単位毎の暗号化コードが正当であることを検証するための検証用情報を格納する手段と、該検証用情報を用いて該単位毎の暗号化コードを検証する手段とを備えることを特徴とするメモリ。

【請求項6】 暗号化コードを格納するメモリから、該暗号化コードが正当であることを示す通知を受け取る手段と、前記通知を受け取ったとき、前記暗号化コードに含まれる暗号化命令をフェッチして復号する手段と、復号された命令を実行する手段とを備えることを特徴とするプロセッサ。

【請求項7】 コンピュータのためのプログラムを記録した記録媒体であって、該プログラムは、

セキュアタスクを実行するセキュアプロセッサと、通常タスクを実行する通常プロセッサを含むマルチプロセッサシステムにおいて、該セキュアタスクと通常タスクの振り分けを行い、

前記セキュアタスクの暗号化コードと、該暗号化コードが正当であることを検証するための検証用情報とを、セキュアメモリに格納し、

前記検証用情報により前記暗号化コードが正当であると検証されたとき、該暗号化コードをセキュアプロセッサに実行させる処理を前記コンピュータに実行させることを特徴とするコンピュータ読み取り可能な記録媒体。

【請求項8】 セキュアタスクを実行するセキュアプロセッサと、通常タスクを実行する通常プロセッサを含むマルチプロセッサシステムにおいて、該セキュアタスクと通常タスクの振り分けを行い、

前記セキュアタスクの暗号化コードと、該暗号化コードが正当であることを検証するための検証用情報とを、セキュアメモリに格納し、

前記検証用情報により前記暗号化コードが正当であると検証されたとき、該暗号化コードをセキュアプロセッサに実行させる処理をコンピュータに実行させるためのプログラム。

【請求項9】 コード作成者が、コード認証機関に実行可能なコードを提供し、

前記コード認証機関が、前記コードが正当であることを検証するための検証用情報を該コードに付加して、マルチプロセッサシステムのユーザに配布し、

前記マルチプロセッサシステムは、前記コードを用いてセキュアタスクを実行するセキュアプロセッサと、通常タスクを実行する通常プロセッサを含み、該セキュアタスクと通常タスクの振り分けを行い、前記検証用情報を用いて該コードが正当であることを検証し、該コードを実行することを特徴とするコード配布方法。

【請求項10】 コード作成者が、コード認証機関に実行可能なコードを提供して、手数料を支払い、

前記コード認証機関が、前記コードが正当であることを検証するための検証用情報を該コードに付加し、

前記コード作成者が、前記コードをマルチプロセッサシステムのユーザに配布して、該ユーザが支払う対価を受領し、

前記マルチプロセッサシステムは、前記コードを用いてセキュアタスクを実行するセキュアプロセッサと、通常タスクを実行する通常プロセッサを含み、該セキュアタスクと通常タスクの振り分けを行い、前記検証用情報を用いて該コードが正当であることを検証し、該コードを実行することを特徴とするコード配布方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、コード化されたプログラムのような、実行可能なコードを実行する装置およびそのようなコードを配布する方法に関する。

【0002】

【従来の技術】電子署名と暗号化が施された実行可能なコード（以下では、単にコードと記す）を、署名を検証した上で復号しながら実行することで、認証されたコードのみが安全に動作する環境が実現できる。実行可能なコードには、コード化されたプログラムの一部または全体が含まれる。このような動作環境の実現方法として、署名検証機能と復号機能を持つプロセッサ（セキュアプロセッサ）をメモリマップドの入出力装置（I/O装置）として見せかける方法が考えられる。この方法では、暗号化コードをデータとしてそのI/O装置に渡し、実行させて、実行結果を取得する。

【0003】

【発明が解決しようとする課題】しかしながら、上述した従来のコード実行方法には、次のような問題がある。このコード実行方法では、I/O装置のメモリ量の制限からサイズの大きなコード全体を一度に渡すことができない。また、一旦コードを渡して実行を開始すると、終わるまで他の制御を行うことができないため、署名検証や復号等のセキュア処理をマルチタスクで実行することができない。このため、セキュア処理を伴う複数のタスクを効率よく実行することができない。

【0004】このうち、後者の問題は、I/O装置をマルチタスク処理が可能なように設計し、そのI/O装置専用のタスク管理モジュールをオペレーティングシステム（OS）内に用意することで解決できる。しかし、OSが、I/O装置（セキュアプロセッサ）専用のタスク管理モジュールを、通常プロセッサ用のタスク管理モジュールと重複して持つことになり、効率上好ましくない。

【0005】さらに、セキュリティ上の観点から、OSそのものがセキュアプロセッサ上で動作することが望ましいが、既存OSをいきなりセキュアプロセッサ用に書き換えることは簡単ではないという問題もある。

【0006】本発明の課題は、既存OSを大幅に変更することなく、電子署名と暗号化が施されたコードを効率よく実行する装置と、そのような装置に対してコードを配布する方法を提供することである。

【0007】

【課題を解決するための手段】図1は、本発明のコード実行装置の原理図である。図1のコード実行装置は、マルチプロセッサシステムを用いて実現され、セキュアメモリ手段11、セキュアプロセッサ手段12、通常メモリ手段13、通常プロセッサ手段14、および制御手段15を備える。

【0008】セキュアメモリ手段11は、セキュアタスクの暗号化コードと、その暗号化コードが正当であるこ

とを検証するための検証用情報とを格納する。セキュアプロセッサ手段12は、検証用情報により暗号化コードが正当であると検証されたとき、その暗号化コードを実行する。通常メモリ手段13は、通常タスクのコードを格納し、通常プロセッサ手段14は、通常タスクのコードを実行する。

【0009】制御手段15は、セキュアタスクと通常タスクの振り分けを行って、暗号化コードをセキュアメモリ手段11に格納し、通常タスクのコードを通常メモリ手段13に格納する。

【0010】検証用情報としては、例えば、電子署名、パリティコード、CRC（Cyclic Redundancy Check）ビットが用いられる。また、制御手段15は、例えば、マルチプロセッサシステムのOSに対応する。

【0011】セキュアタスクを実行する場合、制御手段15は、その暗号化コードを検証用情報とともにセキュアメモリ手段11に格納する。そして、検証用情報を用いて暗号化コードが正当であると検証されると、セキュアプロセッサ手段12が、その暗号化コードを実行する。

【0012】また、通常タスクを実行する場合、制御手段15は、そのコードを通常メモリ手段13に格納する。そして、通常プロセッサ手段14は、そのコードを実行する。

【0013】このように、マルチプロセッサシステム内でセキュアタスクと通常タスクの共存を許し、制御手段15がそれらのタスクをセキュアプロセッサ手段12と通常プロセッサ手段14に振り分けることで、タスク管理が簡単になる。したがって、OSを大幅に変更することなく、セキュアタスクのコードを効率よく実行することができる。

【0014】また、このようなマルチプロセッサシステムに対してコードを配布する方法としては、例えば、以下の2通りの方法がある。

（1）コード作成者が、コード認証機関に実行可能なコードを提供し、コード認証機関が、コードが正当であることを検証するための検証用情報をコードに付加して、マルチプロセッサシステムのユーザに配布する。

（2）コード作成者が、コード認証機関に実行可能なコードを提供して、手数料を支払い、コード認証機関が、検証用情報をコードに付加する。その後、コード作成者が、コードをマルチプロセッサシステムのユーザに配布して、ユーザが支払う対価を受領する。

【0015】

【発明の実施の形態】以下、図面を参照しながら、本発明の実施の形態を詳細に説明する。本実施形態では、マルチプロセッサシステム中の1つのプロセッサをセキュアプロセッサに置き換え、ヘテロなマルチプロセッサシステムを構築する。そして、セキュアなタスクと非セキュアなタスクの各プロセッサへの割り当てを、OSが制

御する。このように、セキュアプロセッサを含むヘテロなマルチプロセッサシステムを構築することで、OSが2種類のタスク管理モジュールを重複して持つ必要がなくなる。

【0016】また、セキュアタスクと非セキュアタスクの共存を許すことで、可能な部分からセキュアタスク化していき、最終的にOSの全タスクをセキュア化するというマイグレーションが可能となる。このようなマイグレーションは、OSが細かなタスクの集合体として実現されているような場合に限られるが、これにより、既存OSをいきなりセキュアプロセッサ用に書き換える必要がなくなる。

【0017】しかし、このようなシステムにおいては、セキュアプロセッサが暗号化コードを1命令ずつ取り出して（フェッチして）実行するため、コード全体をセキュアプロセッサにまとめて渡すことはできない。したがって、暗号化コード全体に署名をつけたのでは、署名の検証ができないという問題が発生する。そこで、本実施形態では、メモリ割り当ての最小単位であるページ（例えば、4Kバイト等）単位に署名を付与した形で暗号化コードを作成し、メモリ割り当て時にメモリ自身が署名を検証する構成を採用する。

【0018】さらに、このシステムに対して配布すべきコードを収集し、収集したコードに署名をつけて配布する機関を設ける。これにより、コード作成者は、コードを広範囲に配布することが可能となり、ユーザは、安心してコードを利用することが可能となる。

【0019】図2は、このようなマルチプロセッサシステムの構成図である。図2のシステムは、通常メモリ21、通常プロセッサ22、セキュアメモリ23、セキュアプロセッサ24、およびセキュアドライブ/メディア25を含む。これらの装置はシステムバス26により互いに接続されているが、通常プロセッサ22は、セキュアメモリ23から命令をフェッチすることではなく、セキュアプロセッサ24は、通常メモリ21から命令をフェッチすることはない。

【0020】通常プロセッサ22は、通常メモリ21を用いて通常のタスク（非セキュアタスク）の通常のコードを実行し、セキュアプロセッサ24は、セキュアメモリ23を用いてセキュアタスクの暗号化コードを実行する。セキュアドライブ/メディア25は、セキュアタスク用の暗号化コードを格納する格納装置である。図2では、通常プロセッサ22とセキュアプロセッサ24が1つずつ設けられているが、各プロセッサを複数設けてもよい。

【0021】図3は、セキュアメモリ23とセキュアプロセッサ24の構成例を示している。図3のセキュアメモリ23は、認証局公開鍵31、署名検証部32、署名保持部33、ページ34を有する。ページ34は、物理メモリ（実メモリ）割り当ての最小単位であり、例え

ば、4Kバイトの容量を持つ。また、署名保持部33は、ページ毎に、ページに対する署名データを格納する領域を持つ。署名検証部32は、例えば、ハードウェアまたはMPU（Micro Processing Unit）を用いて実装され、ページ単位で署名を用いて暗号化コードを検証する。

【0022】署名としては、例えば、認証局（Certificate Authority, CA）の秘密鍵によって作成され、セキュアメモリ23にあらかじめ格納されたCAの公開鍵によって検証可能なX.509証明書が用いられる。

【0023】セキュアプロセッサ24は、復号鍵設定部41、復号鍵保持部42、復号部43、およびプロセッサ44を有する。このうち、復号鍵設定部41、復号鍵保持部42、および復号部43は、命令を実行するプロセッサ44の命令入力部の前段に設けられ、復号部43は、例えば、ハードウェアまたはMPUを用いて実装される。

【0024】署名を含む暗号化コードは、セキュアドライブ/メディア25より読み出され、署名と暗号化コードに分離された後、それぞれ署名保持部33とページ34に格納される。

【0025】これらの格納が終了すると、署名検証部32は、署名を作成した認証局の公開鍵で署名を検証し、問題がなければ、署名に含まれる暗号化コードのハッシュ値を、ページ34上の暗号化コードから再計算したハッシュ値と比較する。

【0026】これらのハッシュ値が一致し、暗号化コードが改竄されていないことを確認すると、ページ34上の暗号化コードが有効である（正当なコードである）ことをセキュアプロセッサ24に通知する。また、ハッシュ値が一致しなければ、ページ34上の暗号化コードが無効である（正当なコードでない）ことをセキュアプロセッサ24に通知する。

【0027】セキュアプロセッサ24の復号部43は、有効の通知を受け取ると、ページ34上のメモリアドレスから必要な暗号化命令をフェッチし、復号鍵保持部42の復号鍵を用いて暗号化命令を順次復号する。そして、プロセッサ44は、復号された命令を順次実行する。復号に必要な復号鍵は、復号鍵設定部41により、あらかじめ復号鍵保持部42に設定されている。

【0028】さらに、セキュアプロセッサ24に複数の復号鍵を格納しておき、どの復号鍵を使用して復号するかを外部より指定可能とすることもできる。図4は、このようなセキュアプロセッサ24の構成図である。

【0029】図4のセキュアプロセッサ24は、複数の復号鍵保持部42を持つ点と、これらの保持部42に格納された複数の復号鍵のうちのどれを使用するかを指示する復号鍵指示部45が追加されている点が、図3のセキュアプロセッサ24とは異なる。復号鍵指示部45は、例えば、ハードウェアまたはMPUを用いて実装さ

れる。どの復号鍵を使用するかは、例えば、現在実行中のセキュアタスクに応じて、OSが復号鍵指示部45に指示する。

【0030】図5は、図2のマルチプロセッサシステムの動作を制御するセキュアOSの構成図である。図5のセキュアOS51は、セキュアプロセッサ24または/および通常プロセッサ22上で動作し、セキュアタスク管理52、セキュアメモリ管理53、およびセキュアファイルシステム54を含む。

【0031】セキュアタスク管理52とセキュアメモリ管理53は、セキュアタスクと非セキュアタスクの振り分けを行う。これにより、セキュアタスクの暗号化コードはセキュアメモリ23に格納され、非セキュアタスクのコードは通常メモリ21に格納される。また、セキュアタスク管理52は、セキュアタスクと非セキュアタスクの両方のマルチタスク処理を制御する。以下、対象とするタスクがセキュアタスクの場合の動作を説明する。

【0032】セキュアタスク管理52は、複数のタスクのコンテキストを管理し、コンテキスト切り替え時に、セキュアプロセッサ24のプログラムカウンタの変更等の通常処理のほか、セキュアプロセッサ24内に保持されている復号鍵のうちどれを使用するかを指示する。

【0033】セキュアメモリ管理53は、必要に応じて、セキュアタスクへのセキュアメモリ23の割り当てを行う。なお、セキュアメモリ23からセキュアプロセッサ24への暗号化命令の転送は、CPU (Central Processing Unit) のフェッチ動作であるので、セキュアメモリ管理53を経由しない。

【0034】セキュアファイルシステム34は、セキュアドライブ/メディア25に格納された暗号化コードのファイルを管理する。そして、セキュアメモリ管理53からの要求に応じて、セキュアドライブ/メディア25から暗号化コードを読み出し、セキュアメモリ管理53に渡す。

【0035】次に、図6から図9までを参照しながら、セキュアタスク管理52、セキュアメモリ管理53、およびセキュアファイルシステム54の処理をより詳細に説明する。

【0036】図6は、セキュアタスク管理52の処理のフローチャートである。図6の処理は、セキュアプロセッサ24上で現在実行中のセキュアタスクのタイムスライスが切れ、タイマ割り込みが発生して、セキュアタスク管理52に制御が移ったときに開始される。

【0037】セキュアタスク管理52は、まず、スケジューリングアルゴリズムに従って、次に実行するセキュアタスクAを決定し(ステップS1)、セキュアタスクAのコンテキストを復元する(ステップS2)。このとき、セキュアプロセッサ24のプログラムカウンタおよびスタックポインタの復元や、セキュアプロセッサ24とセキュアメモリ23の間に存在するMMU (Memory M

anagement Unit) 内のTLB (Translation Look aside Buffer; #65078;) の復元等が行われる。

【0038】次に、プログラム復号鍵としてセキュアタスクA用のものを使用することを、セキュアプロセッサ24に指示する(ステップS3)。そして、セキュアタスクAのタイムスライス(例えば、100ms)をタイマに設定し(ステップS4)、セキュアプロセッサの動作を再開する(ステップS5)。

【0039】図7は、セキュアメモリ管理53の処理のフローチャートである。図7の処理は、セキュアタスク実行中にページフォルトが起き、割り込みが発生して、セキュアメモリ管理53に制御が移ったときに開始される。

【0040】セキュアメモリ管理53は、まず、セキュアメモリ23内に未使用の実メモリ領域があるか否かをチェックし(ステップS11)、実メモリがあれば、それを1ページ割り当てる(ステップS13)。未使用の実メモリがない場合は、実メモリ解放処理のサブルーチン呼び出し(ステップS12)、空きを作った後に実メモリを割り当てる。

【0041】次に、割り当てた実メモリのアドレスと仮想アドレスの対応表を作成し、MMU内のTLBに格納する(ステップS14)。そして、割り当てた実メモリに配置するコードをセキュアファイルシステム54に要求し、受け取ったコードを実メモリに配置して(ステップS15)、セキュアプロセッサ24の動作を再開する(ステップS16)。

【0042】図8は、図7のステップS12において呼び出されたサブルーチンが行う実メモリ解放処理のフローチャートである。サブルーチンは、まず、実メモリ解放アルゴリズムに従って、ページアウトの対象となる実メモリを決定する(ステップS21)。次に、対象となった実メモリ上のコードを、セキュアドライブ/メディア25にページアウトする(書き出す)(ステップS22)。そして、呼び出し元に復帰する(ステップS23)。

【0043】図9は、セキュアファイルシステム54の処理のフローチャートである。図9の処理は、図7のステップS15においてセキュアメモリ管理53からコードの要求を受けたときに開始される。

【0044】セキュアファイルシステム54は、まず、対象となるプログラムの先頭からのオフセットを受け取り(ステップS31)、セキュアドライブ/メディア25内で指定位置までシークする(ステップS32)。そして、指定位置より1ページ分のコードを読み出し、セキュアメモリ管理53に渡す(ステップS33)。

【0045】ところで、セキュアメモリ23とセキュアプロセッサ24が相互認証を行ってセッション鍵を共有することで、暗号化コードをより安全にやり取りすることも可能である。この場合、セキュアメモリ23は、暗

号化コードをセッション鍵でさらに暗号化した上でセキュアプロセッサ24に転送する。

【0046】図10は、このようなセキュアメモリ23とセキュアプロセッサ24の構成図である。図10のセキュアメモリ23は、図3の構成に加えて、さらに相互認証/セッション鍵共有部61および暗号部62を有し、セキュアプロセッサ24は、図4の構成に加えて、さらに相互認証/セッション鍵共有部71および復号部72を有する。

【0047】まず、相互認証/セッション鍵共有部61と相互認証/セッション鍵共有部71は、相互に信頼できる相手であることを認証した上で、セッション鍵を生成/共有する。相互認証の方法は、公開鍵を用いた証明書ベースの方法でもかまわないし、共通鍵を用いた方法でもかまわない。また、セッション鍵は、例えば、乱数を用いて生成される。

【0048】その後、セキュアメモリ23の暗号部62は、ページ34上の暗号化命令をセッション鍵でさらに暗号化して、セキュアプロセッサ24に転送する。セキュアプロセッサ24の復号部72は、受け取った暗号化命令をセッション鍵で復号した後、復号部43に渡す。その後、図4に示したように、暗号化命令が対応する復号鍵で復号されて実行される。

【0049】また、同様にして、セキュアドライブ/メディア25とセキュアメモリ23が相互認証を行ってセッション鍵を共有することで、暗号化コードをより安全にやり取りすることも可能である。

【0050】図11は、このようなセキュアドライブ/メディア25とセキュアメモリ23の構成図である。図11のセキュアドライブ/メディア25は、格納媒体81、格納装置固有鍵82、相互認証/セッション鍵共有部83、復号部84、および暗号部85を有し、セキュアメモリ23は、図10の構成に加えて、さらに復号部63を有する。

【0051】セキュアドライブ/メディア25は、暗号化コードを格納装置固有鍵82または格納媒体固有鍵86でさらに暗号化して、格納媒体81に格納する。格納媒体81としては、磁気ディスク、光ディスク、光磁気ディスク、磁気テープ等が用いられる。また、格納装置固有鍵82は、セキュアドライブ/メディア25固有の鍵であり、格納媒体固有鍵86は、格納媒体81固有の鍵である。

【0052】相互認証/セッション鍵共有部83と相互認証/セッション鍵共有部61は、図10の場合と同様にして、相互に信頼できる相手であることを認証した上で、セッション鍵を生成/共有する。

【0053】セキュアドライブ/メディア25の復号部84は、格納媒体81上に格納された暗号化コード87を、格納装置固有鍵82または格納媒体固有鍵86で復号して、暗号部85に渡す。暗号部85は、相互認証/

セッション鍵共有部83が保持しているセッション鍵により暗号化コードをさらに暗号化して、セキュアメモリ23に転送する。セキュアメモリ23の復号部63は、受け取った暗号化コードをセッション鍵で復号して、元の暗号化コードの形に戻した上で、ページ34に格納する。

【0054】このとき、図5のセキュアファイルシステム54は、セキュアドライブ/メディア25とセキュアメモリ23の間のセッション鍵の共有を仲介する。その後、セキュアファイルシステム54は、このセッション鍵によって暗号化された暗号化コードを、格納媒体81上の論理フォーマットに従ってセキュアドライブ/メディア25から読み出し、セキュアメモリ23に転送する。

【0055】ここで、1つのセキュアタスクが実行される場合の処理の流れを説明する。この場合、セキュアタスクには現在セキュアメモリが1ページだけ割り当てられており、プログラムカウンタはそのページ上の暗号化コードの最後の暗号化命令を指しているものとする。また、各エンティティ（セキュアメモリ23、セキュアプロセッサ24、セキュアドライブ/メディア25）間の相互認証処理、セッション鍵の共有処理、セッション鍵による暗号化/復号処理については、説明を省略する。

(1) セキュアプロセッサ24は、セキュアメモリ23より暗号化コードをフェッチし、復号した後に実行する。

(2) セキュアプロセッサ24は、プログラムカウンタをインクリメントし、次の命令のフェッチ動作を実行する。

(3) 実メモリが未割り当てであるため、セキュアメモリ23は、ページフォルト例外をセキュアタスク管理52に対して発生する。

(4) セキュアタスク管理52は、実行中のセキュアタスクをスリープ状態に設定した後、セキュアメモリ管理53に新たな実メモリの割り当てを依頼する。

(5) セキュアメモリ管理53は、新たな実メモリ1ページをセキュアタスクに割り当てる。

(6) セキュアタスク管理52は、暗号化コードの続きを読み出すように、セキュアファイルシステム54に依頼する。

(7) セキュアファイルシステム54は、暗号化コードの続きをセキュアドライブ/メディア25より読み出し、新たに割り当てられた実メモリに格納する。

(8) セキュアタスク管理52は、スリープ状態のセキュアタスクを実行状態に設定する。

(9) セキュアプロセッサ24は、新たに割り当てられたページ上の次の命令をフェッチして実行する。

【0056】次に、2つのセキュアタスクA、Bが実行される場合の処理の流れを説明する。この場合、セキュアタスクAおよびBともに、十分なセキュアメモリが割

り当てられており、ページフォルトは起きないものとする。

(1) セキュアプロセッサ24は、セキュアタスクAの暗号化命令をフェッチして実行する。

(2) セキュアタスク管理52は、タイムスライスが切れ、タイマ割り込みが発生したため、セキュアタスクAをスリープ状態に設定する。

(3) セキュアタスク管理52は、スケジューリングアルゴリズムに従って、次に動作させるタスクをセキュアタスクBに決定し、セキュアタスクBを動作状態に設定する。

(4) セキュアタスク管理52は、セキュアタスクBの復号に必要な鍵をセキュアプロセッサ24に対して指示する。

(5) セキュアタスク管理52は、プログラムカウンタ、スタックポインタ、TLBのアドレス対応表等をセキュアタスクB用に設定する。

(6) セキュアプロセッサ24は、セキュアタスクBの暗号化命令をフェッチして実行する。

【0057】以上の説明は、セキュアOSがセキュアプロセッサ24上で動作するものと考えると容易に理解できる。しかし、セキュアプロセッサ24の実行を一時停止する機能や、セキュアプロセッサ24のプログラムカウンタの変更等のようなコンテキストを切り替える機能がセキュアプロセッサ24に用意されていれば、セキュアOS自身は通常プロセッサ22上で動作させることも可能である。

【0058】図2のマルチプロセッサシステムでは、セキュアメモリ23と通常メモリ21が別々に設けられているが、セキュアメモリ23と通常メモリ21の一部または全部がオーバーラップした形態も考えられる。

【0059】図12および図13は、このようなマルチプロセッサシステムの構成例を示している。ただし、ここではセキュアドライブ/メディア25は省略されている。図12において、セキュアプロセッサ24と通常プロセッサ22は同一のシステムバス92（データバス、アドレスバス）を介して、セキュアメモリ91に接続されている。この場合、セキュアメモリ91は、図2のセキュアメモリ23と通常メモリ21の機能を兼ね備えている。

【0060】また、図13において、セキュアプロセッサ24は、システムバス94を介して固有のセキュアメモリ23に接続されており、システムバス95を介して共有メモリ93に接続されている。また、通常プロセッサ22は、システムバス96を介して固有の通常メモリ21に接続されており、システムバス95を介して共有メモリ93に接続されている。共有メモリ93は、セキュアプロセッサ24と通常プロセッサ22に共通のメモリであり、セキュアメモリ23または／および通常メモリ21の機能を備えている。

【0061】図12の構成は、システムバスおよびメモリが1つずつしかないため、図13の構成よりコストが低いという利点がある。しかし、セキュアプロセッサ24と通常プロセッサ22の両方がセキュアメモリ91にアクセス可能なため、図13の構成よりセキュリティレベルが低くなる。逆に、図13の構成は、図12の構成よりコストが増加するが、セキュリティレベルも向上する。

【0062】以上の実施形態では、セキュアプロセッサ24がコードをフェッチして実行しているが、コードの一部または全部を用いて、暗号化命令をフェッチし復号した上で実行する論理回路を自動的に生成することも考えられる。この場合、汎用論理回路を特定の回路状態に固定する装置がシステム内に設けられる。

【0063】セキュアメモリ23が正当なコードであることを検証した後、セキュアプロセッサ24は、そのコードを用いて論理回路の一部または全部を、回路状態として不揮発的に固定する。このとき、前の回路状態を消去して、新しく上書きする。

【0064】図14は、このような回路生成処理のフローチャートである。セキュアプロセッサ24は、まず、暗号化命令をフェッチして復号し（ステップS41）、コードを演算処理回路構成情報に翻訳する（ステップS42）。次に、回路構成情報を配線情報に翻訳し（ステップS43）、配線情報を不揮発的に焼き付ける（ステップS44）。配線情報の焼き付け方法としては、例えば、以下の2通りが考えられる。

(1) 図15に示すように、複数の基本回路を配列（アレイ）状に並べ、配線情報に基づいて回路間を不揮発的に接続し、演算器を構成する。

(2) 図16に示すように、構成済みの基本演算器を多種用意しておき、配線情報に基づいて必要な演算器間を不揮発的に接続する。

【0065】このように、処理部分をハードウェア化することで処理速度が向上する。また、ハードウェアとソフトウェア処理を併用すれば、暗号化命令を階層化して、セキュリティレベルを向上させることもできる。例えば、特に重要な部分の命令は、厳重な認証ステップを経てハードウェア化しておき、それ以外の命令は、ユーザの利便性を図るため、軽い認証で毎回ソフトウェア処理する。

【0066】以上の実施形態では、署名を用いてコードが正当か否かが検証されているが、コードが正当であることを検証するための情報（検証用情報）としては、他の任意の情報を用いることができる。例えば、パリティコード、CRC（Cyclic Redundancy Check）ビット等を付加しておくことで、コードが壊れているか否かを検証することができる。そこで、以下では、署名の代わりに検証用情報という言葉を用い、この情報をコードに付加する機関をコード認証機関と呼ぶことにする。

【0067】次に、図17から図22までを参照しながら、検証用情報が付加されたコードの配布方法について説明する。図17は、ユーザへのコード配布方法を示している。図17において、コード作成者101は、コード認証機関102にコードを提供する(P1)。コード認証機関102は、受け取ったコードの正当性を確認した上で、検証用情報を付加し、認証済みコードをコード利用者103(ユーザ)に提供する(P2)。コード利用者103は、例えば、上述したマルチプロセッサシステムを保有しており、受け取ったコードに付加された検証用情報を用いてコードの正当性を確認した後に、そのコードを利用する。

【0068】このとき、コード認証機関102は、コード作成者101に対して対価を提示してコードを収集し、収集に際して対価を支払う。そして、コード利用者103に対してコードの対価を提示し、検証用情報を付加した後、コード利用者103に対してコードを提供すると同時に対価を徴収する。

【0069】図18は、このような対価の支払いを示している。図18において、コード作成者101は、コード認証機関102にコードを提供し(P11)、その対価をコード認証機関102から受け取る(P12)。コード認証機関102は、コード利用者103に認証済みコードを提供し(P13)、コード利用者103は、それに対する対価をコード認証機関102に支払う(P14)。

【0070】コード利用者103およびコード認証機関102が支払う対価は、コード提供時に一度に課金されてもかまわないし、コードの利用/提供状況に応じて従量課金されてもかまわない。後者の場合、例えば、コード利用者103が受領したコード数に応じて課金が行われる。

【0071】また、コード作成者101がコード認証機関102に手数料を支払うことで、コードに検証用情報を付加してもらい、コード利用者103が支払う対価を受領することも可能である。

【0072】図19は、このようなコード配布方法を示している。図19において、コード作成者101は、コード認証機関102にコードを提供し(P21)、検証用情報を付加してもらうのに必要な手数料を支払って(P22)、認証済みコードを取得する(P23)。次に、認証済みコードをコード利用者103に提供して(P24)、その対価を受け取る(P25)。

【0073】コード利用者103が支払う対価は、コード提供時に一度に課金されてもかまわないし、コードの利用/提供状況に応じて従量課金されてもかまわない。また、コード作成者101が支払う手数料も、同様に、一括課金でもかまわないし、従量課金でもかまわない。

【0074】また、コード作成者101の代わりに、コード認証機関102がコードを配布してもよい。この場

合、コード認証機関102が、認証済みコードをコード利用者103に提供して対価を徴収し、徴収した対価をコード作成者101に支払う。

【0075】また、図17のコード配布方法において、コード認証機関102は、受け取ったコードを2つ以上の部分に分割し、最初に一部を配布し、その後、コード利用者103の要求に応じて、残りの部分を配布することも可能である。この場合、最初の配布は、例えば、以下のいずれかの方法で行われる。

(1) 複数のユーザにコードを放送する。

(2) 各ユーザに、ネットワーク上からコードを自由にダウンロードさせる。

(3) コードを可搬記録媒体に収納して、その記録媒体をユーザに配布する。

【0076】図20は、このようなコード配布方法を示している。図20において、コード作成者101は、コード認証機関102にコードを提供する(P31)。コード認証機関102は、コードの正当性を確認した後、検証用情報を付加して、コード利用者103に認証済みコードの一部を提供する(P32)。コード利用者103は、提供された一部のコードが正当であることを検証用情報を用いて確認した後、そのコードを利用する。さらに、必要であれば、残りの認証済みコードをコード認証機関102より入手して利用する(P33)。

【0077】最初に提供されるコードは、例えば、印刷機能に制限を設けた年賀状作成ソフトや、最初の画面データのみを収録したゲームソフト等である。残りのコードは、例えば、すべての機能制限を取り除いた年賀状作成ソフトや、第2画面以降を収録したゲームソフト等である。

【0078】このとき、コード認証機関102は、コード作成者101に対して対価を提示してコードを収集し、収集に際して対価を支払う。次に、コード利用者103に対して残しておいた部分のコードの対価を提示し、検証用情報を付加した後、コードを提供すると同時に対価を徴収する。

【0079】図21は、このような対価の支払いを示している。図21において、P41およびP42の処理は、図18のP11およびP12の処理と同様である。次に、コード認証機関102は、認証済みコードの一部を、例えば、雑誌付録のCD-ROM(compact disk read only memory)やインターネットを通じて、無償で配布する(P43)。これを入手して利用したコード利用者103は、さらに残りのコードを利用したい場合には、コード認証機関102に対価を支払い(P45)、残りのコードの提供を受ける(P44)。

【0080】また、図19と同様に、コード作成者101がコード認証機関102に手数料を支払うことで、コードに検証用情報を付加してもらうことも可能である。この場合、コード作成者101がコード利用者103に

対して残りの部分のコードの対価を提示し、コード利用者103に対してコードを提供すると同時に対価を徴収する。

【0081】図22は、このようなコード配布方法を示している。図22において、P51、P52、およびP53の処理は、図19のP21、P22、およびP23の処理と同様である。次に、コード作成者101は、認証済みコードの一部を、例えば、上述したような方法で無償配布する(P54)。これを入手して利用したコード利用者103が残りのコードを利用したい場合には、対価を支払い(P56)、残りのコードを入手する(P55)。

【0082】また、コード作成者101の代わりに、コード認証機関102がコードを配布してもよい。この場合、コード認証機関102が、コード利用者103に対して残りの部分のコードの対価を提示し、コード利用者103に対してコードを提供すると同時に対価を徴収し、徴収した対価をコード作成者101に支払う。

【0083】以上説明したようなコード配布方法によれば、コード認証機関による認証済みのコードが配布されるので、ユーザは、安心してコードを利用することができる。これにより、コードを利用するユーザが増加し、コードを広範囲に配布することが可能となる。

【0084】ところで、図5のセキュアOS51は、例えば、セキュアドライブ/メディア25にあらかじめ格納され、必要に応じてメモリにロードされて、動作を開始する。また、セキュアOS51を外部に保存しておき、必要に応じてシステムにインストールすることも可能である。

【0085】図23は、セキュアOS51を含むプログラムとデータをマルチプロセッサシステムに供給することのできるコンピュータ読み取り可能な記録媒体を示している。

【0086】サーバ111のデータベース112や可搬記録媒体113に保存されたプログラムとデータは、マルチプロセッサシステムのメモリ114にロードされる。このとき、サーバ111は、プログラムとデータを搬送する搬送信号を生成し、ネットワーク上の任意の伝送媒体を介してマルチプロセッサシステムに送信する。そして、マルチプロセッサシステムは、そのデータを用いてそのプログラムを実行し、必要な処理を行う。

【0087】可搬記録媒体113としては、メモリカード、フロッピー（登録商標）ディスク、CD-ROM、光ディスク、光磁気ディスク等、任意のコンピュータ読み取り可能な記録媒体が用いられる。また、メモリ114は、図2の通常メモリ21またはセキュアメモリ23、図12のセキュアメモリ91、あるいは図13の共有メモリ93に対応する。

（付記1） マルチプロセッサシステムを用いたコード実行装置であって、セキュアタスクの暗号化コードと、

該暗号化コードが正当であることを検証するための検証用情報とを格納するセキュアメモリ手段と、前記検証用情報により前記暗号化コードが正当であると検証されたとき、該暗号化コードを実行するセキュアプロセッサ手段と、通常タスクのコードを格納する通常メモリ手段と、前記通常タスクのコードを実行する通常プロセッサ手段と、前記セキュアタスクと通常タスクの振り分けを行って、前記暗号化コードを前記セキュアメモリ手段に格納し、前記通常タスクのコードを前記通常メモリ手段に格納する制御手段とを備えることを特徴とするコード実行装置。

（付記2） 前記セキュアメモリ手段は、物理メモリ割り当ての単位毎に前記暗号化コードを格納し、該単位毎の暗号化コードに対する検証用情報を格納し、該検証用情報を用いて該単位毎の暗号化コードを検証し、前記セキュアプロセッサ手段は、正当であると検証された暗号化コードに含まれる暗号化命令をフェッチし、復号して実行することを特徴とする付記1記載のコード実行装置。

（付記3） 前記セキュアプロセッサ手段は、複数の復号鍵を保持し、該複数の復号鍵のうち指示された復号鍵を用いて、前記暗号化命令を復号することを特徴とする付記2記載のコード実行装置。

（付記4） 前記セキュアメモリ手段とセキュアプロセッサ手段は、相互認証を行ってセッション鍵を共有し、該セキュアメモリ手段は、前記暗号化命令を該セッション鍵でさらに暗号化して該セキュアプロセッサ手段に転送することを特徴とする付記2記載のコード実行装置。

（付記5） 前記暗号化コードを固有鍵でさらに暗号化して格納するセキュアドライブ手段をさらに備え、該セキュアドライブ手段と前記セキュアメモリ手段は、相互認証を行ってセッション鍵を共有し、該セキュアドライブ手段は、前記制御手段からの読み出し指示に基づいて、該暗号化コードを該固有鍵で復号し、該セッション鍵で暗号化して、該セキュアメモリ手段に転送することを特徴とする付記1記載のコード実行装置。

（付記6） 前記セキュアメモリ手段と通常メモリ手段の領域の少なくとも一部がオーバーラップしていることを特徴とする付記1記載のコード実行装置。

（付記7） 前記セキュアプロセッサ手段は、前記暗号化コードを用いて、該暗号化コードを実行する論理回路の少なくとも一部を、回路状態として不揮発的に固定することを特徴とする付記1記載のコード実行装置。

（付記8） 前記セキュアプロセッサ手段は、前記論理回路の前の回路状態を消去して、新しく上書きすることを特徴とする付記7記載のコード実行装置。

（付記9） 物理メモリ割り当ての単位毎に暗号化コードを格納する手段と、該単位毎の暗号化コードが正当であることを検証するための検証用情報を格納する手段と、該検証用情報を用いて該単位毎の暗号化コードを検

証する手段とを備えることを特徴とするメモリ。

(付記10) 暗号化コードを格納するメモリから、該暗号化コードが正当であることを示す通知を受け取る手段と、前記通知を受け取ったとき、前記暗号化コードに含まれる暗号化命令をフェッチして復号する手段と、復号された命令を実行する手段とを備えることを特徴とするプロセッサ。

(付記11) コンピュータのためのプログラムを記録した記録媒体であって、該プログラムは、セキュアタスクを実行するセキュアプロセッサと、通常タスクを実行する通常プロセッサを含むマルチプロセッサシステムにおいて、該セキュアタスクと通常タスクの振り分けを行い、前記セキュアタスクの暗号化コードと、該暗号化コードが正当であることを検証するための検証用情報とを、セキュアメモリに格納し、前記検証用情報により前記暗号化コードが正当であると検証されたとき、該暗号化コードをセキュアプロセッサに実行させる処理を前記コンピュータに実行させることを特徴とするコンピュータ読み取り可能な記録媒体。

(付記12) セキュアタスクを実行するセキュアプロセッサと、通常タスクを実行する通常プロセッサを含むマルチプロセッサシステムにおいて、該セキュアタスクと通常タスクの振り分けを行い、前記セキュアタスクの暗号化コードと、該暗号化コードが正当であることを検証するための検証用情報とを、セキュアメモリに格納し、前記検証用情報により前記暗号化コードが正当であると検証されたとき、該暗号化コードをセキュアプロセッサに実行させる処理をコンピュータに実行させるためのプログラム。

(付記13) コンピュータのためのプログラムを該コンピュータに搬送する搬送信号であって、該プログラムは、セキュアタスクを実行するセキュアプロセッサと、通常タスクを実行する通常プロセッサを含むマルチプロセッサシステムにおいて、該セキュアタスクと通常タスクの振り分けを行い、前記セキュアタスクの暗号化コードと、該暗号化コードが正当であることを検証するための検証用情報とを、セキュアメモリに格納し、前記検証用情報により前記暗号化コードが正当であると検証されたとき、該暗号化コードをセキュアプロセッサに実行させる処理を前記コンピュータに実行させることを特徴とする搬送信号。

(付記14) コード作成者が、コード認証機関に実行可能なコードを提供し、前記コード認証機関が、前記コードが正当であることを検証するための検証用情報を該コードに付加して、マルチプロセッサシステムのユーザに配布し、前記マルチプロセッサシステムは、前記コードを用いてセキュアタスクを実行するセキュアプロセッサと、通常タスクを実行する通常プロセッサを含み、該セキュアタスクと通常タスクの振り分けを行い、前記検証用情報を用いて該コードが正当であることを検証し、

該コードを実行することを特徴とするコード配布方法。

(付記15) 前記コード認証機関は、前記コード作成者に対して対価を提示して前記コードを収集し、収集に際して対価を支払い、前記ユーザに対して該コードの対価を提示し、前記検証用情報を付加した後、該ユーザに対して該コードを提供すると同時に対価を徴収することを特徴とする付記14記載のコード配布方法。

(付記16) 前記コード認証機関は、前記コードを2つ以上の部分に分割し、最初に一部を配布し、その後、前記ユーザの要求に応じて、残りの部分を配布することを特徴とする付記14記載のコード配布方法。

(付記17) 前記コード認証機関は、前記コード作成者に対して対価を提示してコードを収集し、収集に際して対価を支払い、前記ユーザに対して前記残りの部分の対価を提示し、検証用情報を付加した後、コードを提供して対価を受領することを特徴とする付記16記載のコード配布方法。

(付記18) コード作成者が、コード認証機関に実行可能なコードを提供して、手数料を支払い、前記コード認証機関が、前記コードが正当であることを検証するための検証用情報を該コードに付加し、前記コード作成者が、前記コードをマルチプロセッサシステムのユーザに配布して、該ユーザが支払う対価を受領し、前記マルチプロセッサシステムは、前記コードを用いてセキュアタスクを実行するセキュアプロセッサと、通常タスクを実行する通常プロセッサを含み、該セキュアタスクと通常タスクの振り分けを行い、前記検証用情報を用いて該コードが正当であることを検証し、該コードを実行することを特徴とするコード配布方法。

(付記19) 前記コード作成者は、前記コードを2つ以上の部分に分割し、最初に一部を配布し、その後、前記ユーザの要求に応じて、残りの部分の対価を提示し、コードを提供して対価を受領することを特徴とする付記18記載のコード配布方法。

【0088】

【発明の効果】本発明によれば、セキュアプロセッサを含むヘテロなマルチプロセッサシステムを構築して、セキュアタスクと非セキュアタスクを振り分けることで、OSによる制御が簡単になり、セキュア処理を効率よく実行することができる。また、セキュアタスクのコードに署名を施す際に、メモリに読み込む部分毎に署名を付与することで、コードを効率よく実行することが可能となる。

【図面の簡単な説明】

【図1】本発明のコード実行装置の原理図である。

【図2】第1のマルチプロセッサシステムの構成図である。

【図3】セキュアメモリとセキュアプロセッサの第1の構成図である。

【図4】セキュアプロセッサの構成図である。

【図5】セキュアOSの構成図である。

【図6】セキュアタスク管理の処理のフローチャートである。

【図7】セキュアメモリ管理の処理のフローチャートである。

【図8】実メモリ解放処理のフローチャートである。

【図9】セキュアファイルシステムの処理のフローチャートである。

【図10】セキュアメモリとセキュアプロセッサの第2の構成図である。

【図11】セキュアドライブ／メディアとセキュアメモリの構成図である。

【図12】第2のマルチプロセッサシステムの構成図である。

【図13】第3のマルチプロセッサシステムの構成図である。

【図14】回路生成処理のフローチャートである。

【図15】基本回路の配列を示す図である。

【図16】演算器群を示す図である。

【図17】第1のコード配布方法を示す図である。

【図18】第1の対価の支払いを示す図である。

【図19】第2のコード配布方法を示す図である。

【図20】第3のコード配布方法を示す図である。

【図21】第2の対価の支払いを示す図である。

【図22】第4のコード配布方法を示す図である。

【図23】記録媒体を示す図である。

【符号の説明】

- 11 セキュアメモリ手段
- 12 セキュアプロセッサ手段
- 13 通常メモリ手段
- 14 通常プロセッサ手段
- 15 制御手段
- 21 通常メモリ

- 22 通常プロセッサ
- 23、91 セキュアメモリ
- 24 セキュアプロセッサ
- 25 セキュアドライブ／メディア
- 26、92、94、95、96 システムバス
- 31 認証局公開鍵
- 32 署名検証部
- 33 署名保持部
- 34 ページ
- 41 復号鍵設定部
- 42 復号鍵保持部
- 43、63、72、84 復号部
- 44 プロセッサ
- 45 復号鍵指示部
- 51 セキュアOS
- 52 セキュアタスク管理
- 53 セキュアメモリ管理
- 54 セキュアファイルシステム
- 61、71、83 相互認証／セッション鍵共有部
- 62、85 暗号部
- 81 格納媒体
- 82 格納装置固有鍵
- 86 格納媒体固有鍵
- 87 暗号化コード
- 93 共有メモリ
- 101 コード作成者
- 102 コード認証機関
- 103 コード利用者
- 111 サーバ
- 112 データベース
- 113 可搬記録媒体
- 114 メモリ

【図1】

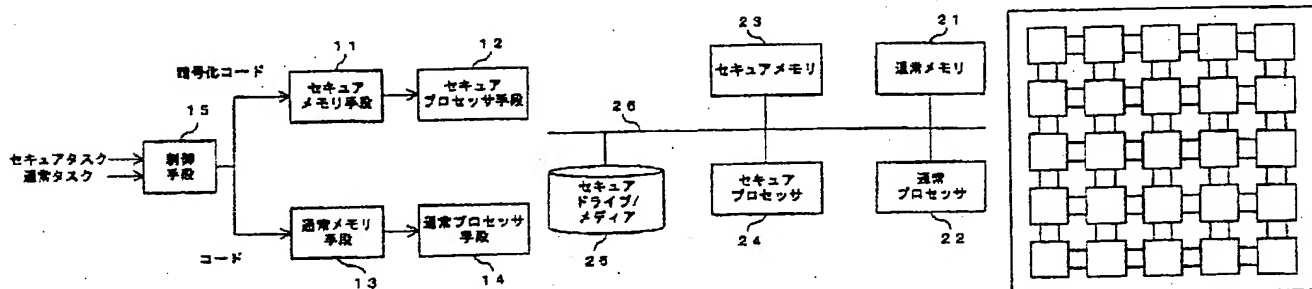
【図2】

【図15】

本 発 明 の 原 理 図

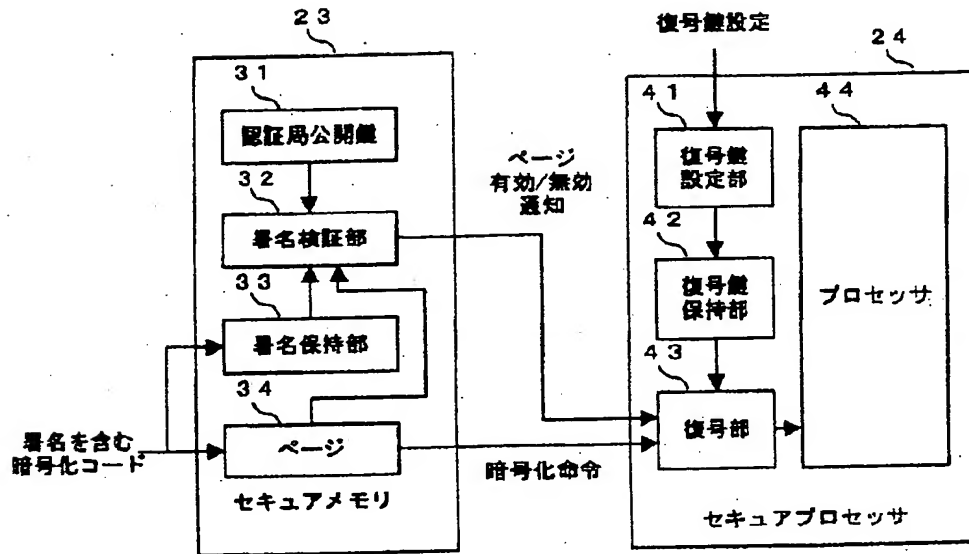
第1のマルチプロセッサシステムの構成図

基本回路の配列を示す図



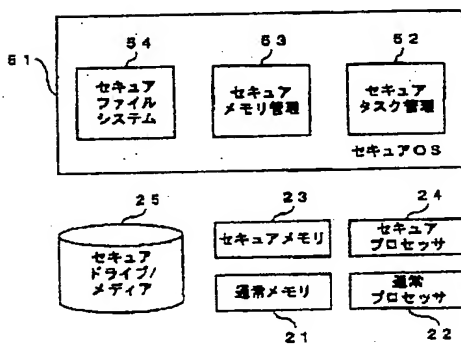
【図3】

セキュアメモリとセキュアプロセッサの第1の構成図



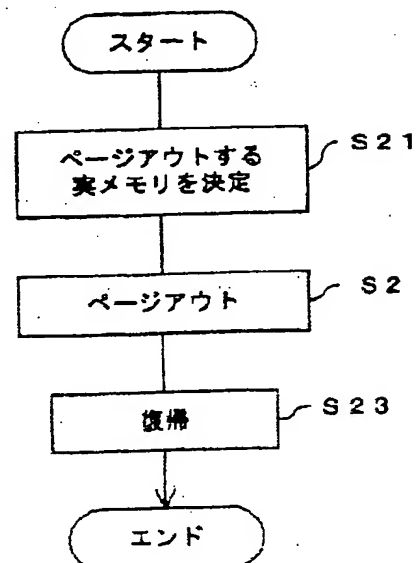
【図5】

セキュアOSの構成図



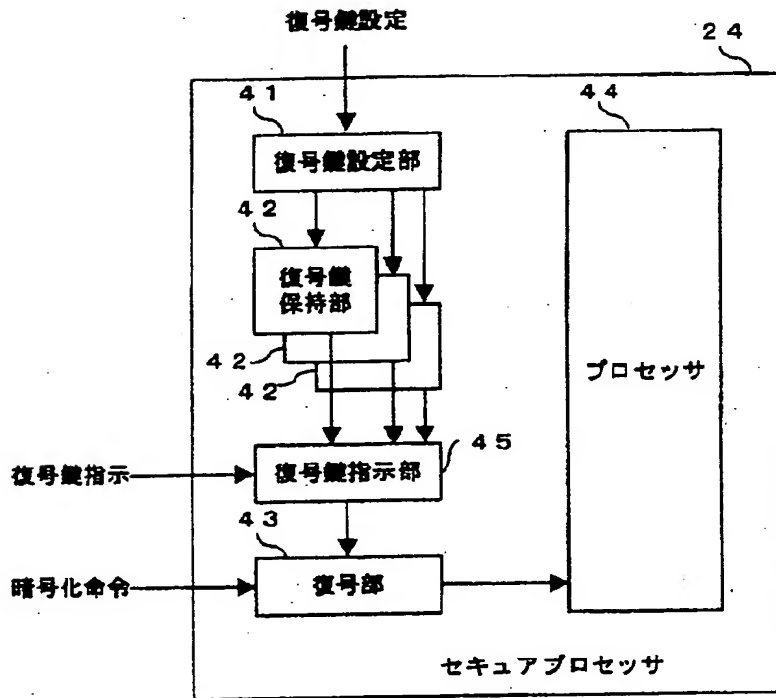
【図8】

実メモリ解放処理のフローチャート



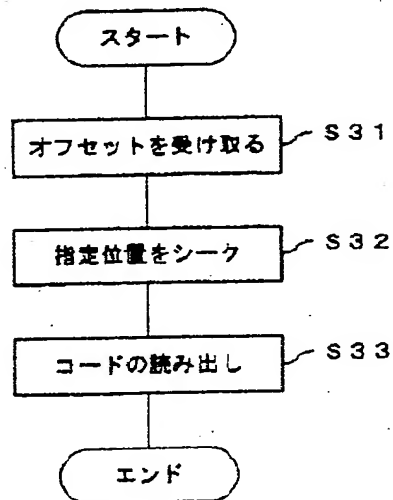
【図4】

セキュアプロセッサの構成図



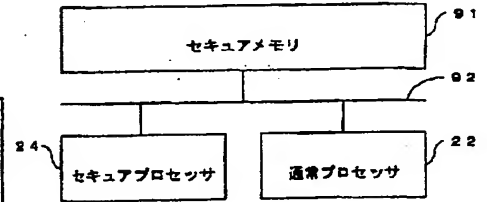
【図9】

セキュアファイルシステムの処理のフローチャート



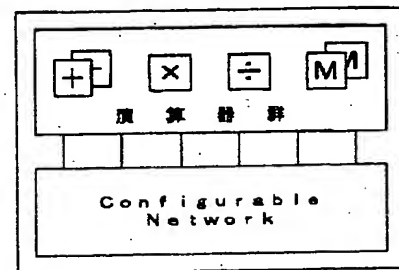
【図12】

第2のマルチプロセッサシステムの構成図



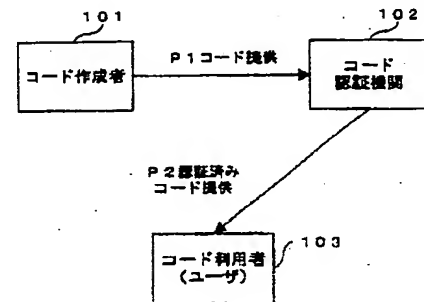
【図16】

演算器群を示す図



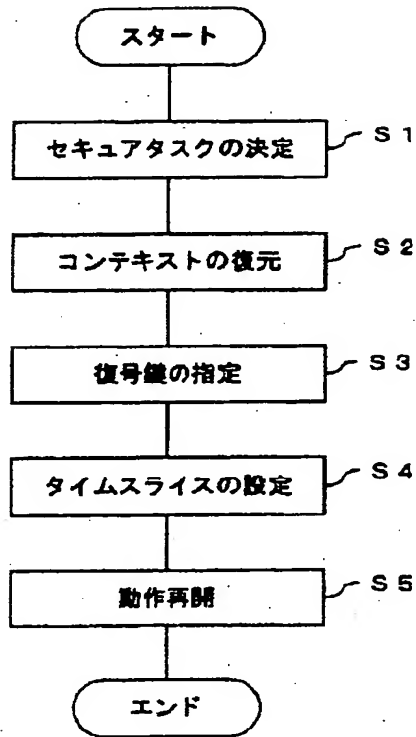
【図17】

第1のコード配布方法を示す図



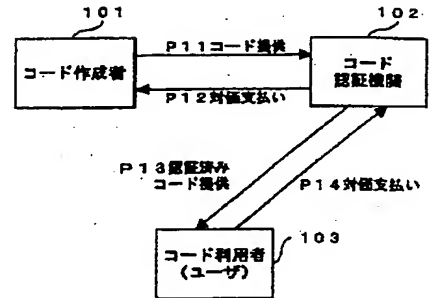
【図6】

セキュアタスク管理の処理へのフローチャート



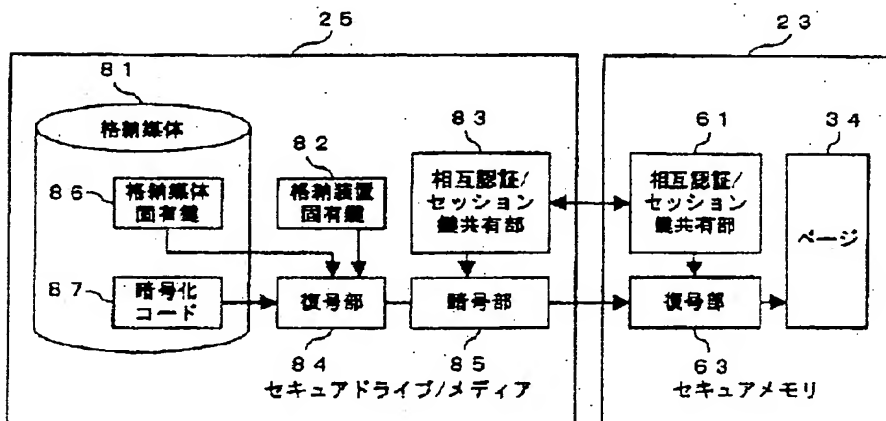
【図18】

第1の対価の支払いを示す図



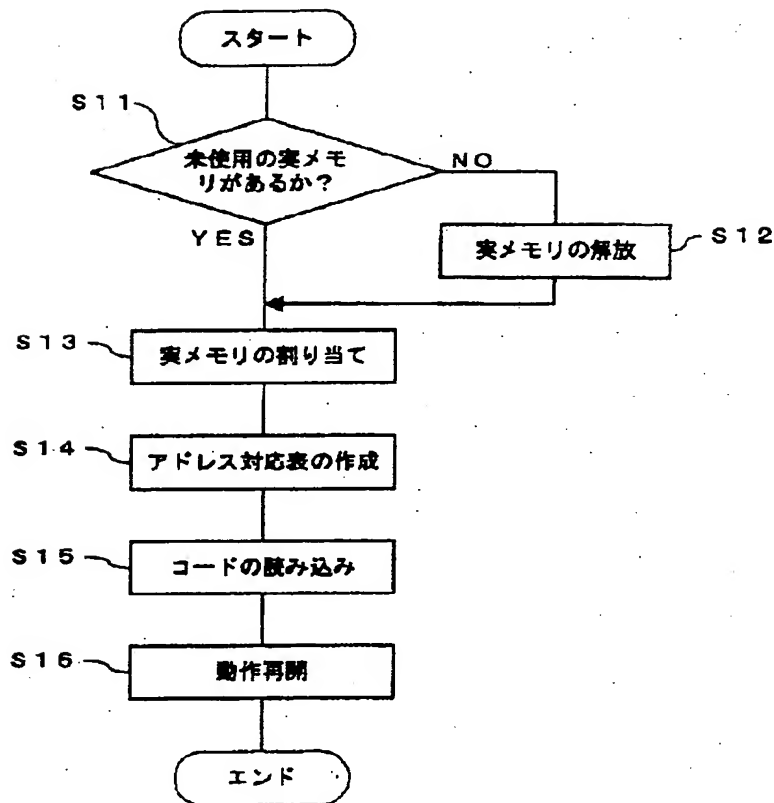
【図11】

セキュアドライブ/メディアとセキュアメモリの構成図



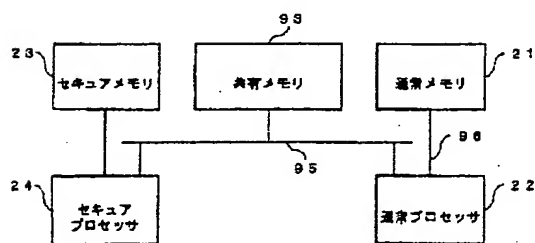
【図7】

セキュアメモリ管理の処理のフローチャート



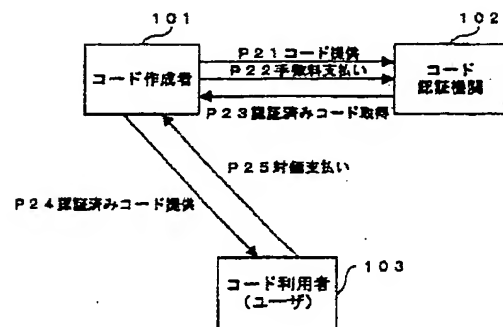
【図13】

第3のマルチプロセッサシステムの構成図



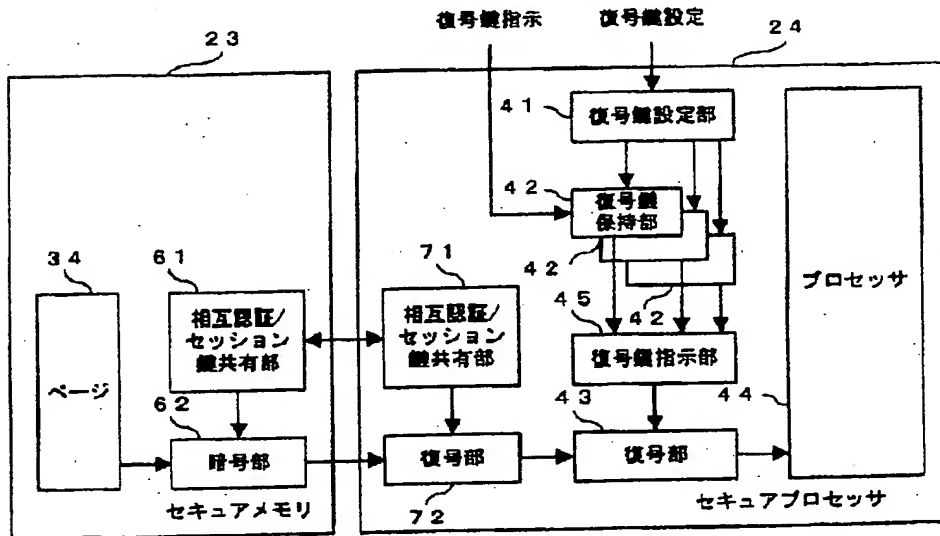
【図19】

第2のコード配布方法を示す図



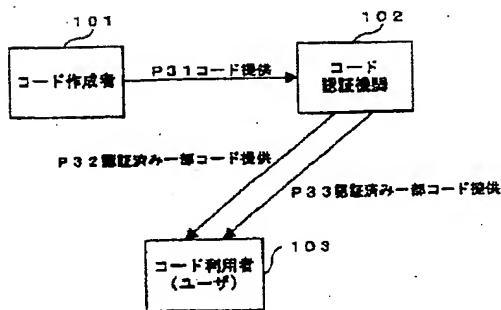
【図10】

セキュアメモリとセキュアプロセッサの第2の構成図



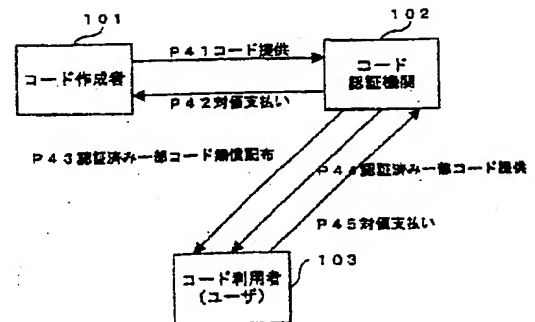
【図20】

第3のコード配布方法を示す図



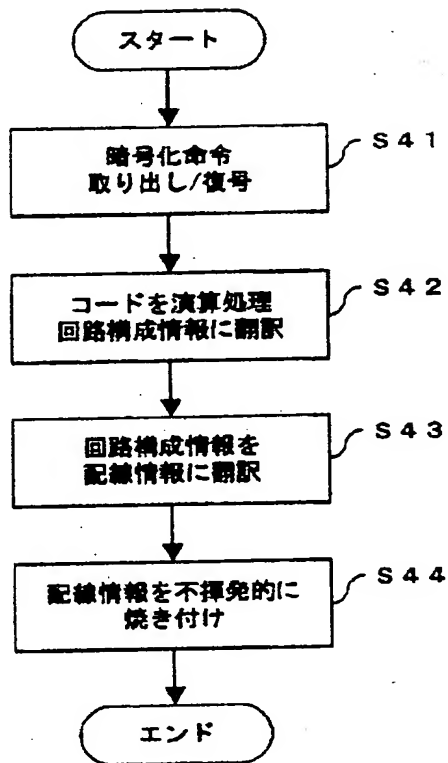
【図21】

第2の対価の支払いを示す図



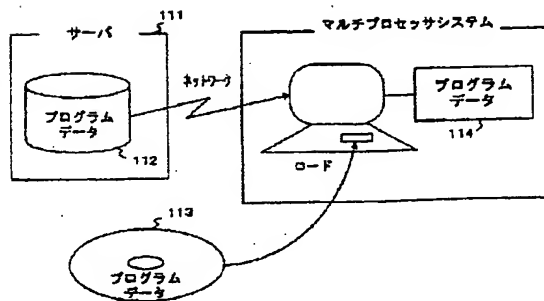
【図14】

回路生成処理のフローチャート



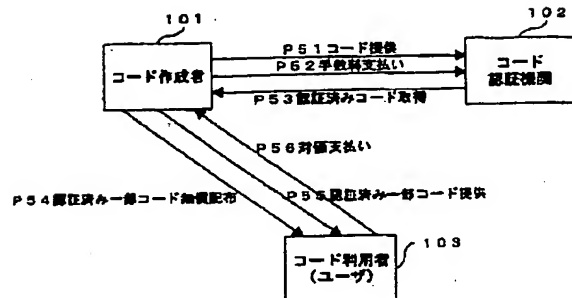
【図23】

記録媒体を示す図



【図22】

第4のコード配布方法を示す図



フロントページの続き

(51)Int.Cl.

G06F 15/16

H04L 9/08

識別記号

610

FI

G06F 9/06

9/30

テーマコード(参考)

660G 5J104

320C

Fターム(参考) 5B017 AA03 BA07 CA15

5B033 BB03

5B045 DD01 GG06 GG09

5B076 FA01 FD04

5B098 AA10 GA04 GC01

5J104 AA09 EA06 LA03 NA02